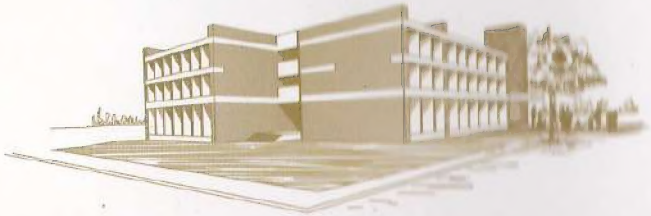
Question
Explore
Discover



Vikram A Sarabhai Community Science Centre (VASCSC)

VASCSC is a pioneering institution working in the field of science education, founded by Dr. Vikram Sarabhai in 1966. Its mandate is to stimulate interest, encourage and expose the principles of science and scientific method in the community and also to improve and innovate various areas of science education. VASCSC has well-equipped laboratories in Biology, Chemistry, Physics, Computers, Electronics, Mathematics and Model Rocketry as well as a Workshop, Library and Innovation Hub.

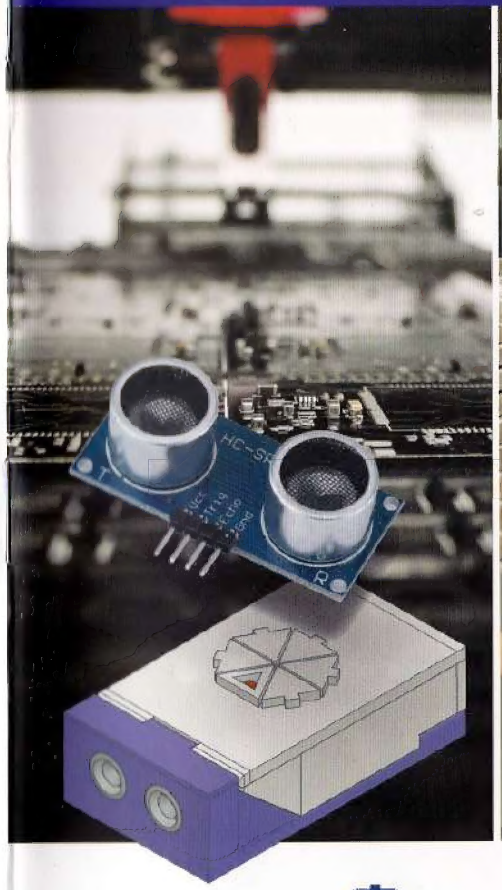
VIKRAM A SARABHAI
COMMUNITY SCIENCE CENTRE

Navrangpura, Ahmedabad - 380 009
Tel: +91 79 26302914, 26302085
E: vascsc@gmail.com W: www.vascsc.org

Follow us on

ULTRASONIC & IR SENSORS

Student Projects



VIKRAM A SARABHAI
COMMUNITY SCIENCE CENTRE

Guidance : Dilip Surkar
Compilation : Harsh Bhatt, Mohit Ahuja
Editing : Neelam Mishra, Harsh Bhatt
Design and Layout : Chetan Patel

First Edition : 2021
ISBN : 978-93-80580-30-2
Price : ₹ 60/-

VIKRAM A SARABHAI COMMUNITY SCIENCE CENTRE
Navrangpura, Ahmedabad - 380 009
Tel: +91 79 26302914, 26302085
E: vascsc@gmail.com W: www.vascsc.org

Table of Content

Ultrasonic Sensor

IR Sensor

Ultrasonic Sensor

1. Getting to know Ultrasonic Sensor

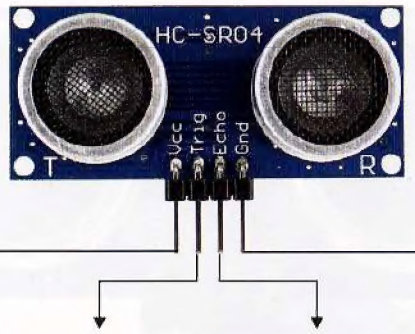
This is one of the most frequently used sensors in a makerspace/ATL. A small list of things that it can sense include:

Obstacle **Distance** **Height** **Level** **Velocity**

Technical Specifications

Name:	HC-SR04 Ultrasonic Rangefinder
Operating Voltage:	5 volts
Average Current Consumption:	15mA
Ultrasonic Frequency:	40 Khz
Range of sensing (Theoretical):	2cm - 400cm
Angle of Measurement	15 degrees
Dimensions (lxbxh):	43cm x 20cm x 15cm

Pin Information



VCC Pin +ve Pin 5V Pin	Trig Pin Digital Output Pin	Echo Pin Digital Input Pin	GND Pin -ve Pin 0V Pin
This pin is to power up the sensor. Typically, a sensor needs 5V to work properly. So, this pin needs to be connected with a 5V of microcontroller or regulated external power supply of 5V.	This pin connects to the digital I/O pin of the microcontroller and creates pulses of Ultrasonic sound. These pulses then strike objects in their path and create reflected pulses.	This pin also connects to the digital I/O pin of the microcontroller. This pin records the reflected pulses and sends them to the microcontroller for processing.	It is a common pin which need to be connected with Ground pin of the microcontroller

2. Serial Radar

Project 1

This example shows the distance measured by the sensor on the Serial Monitor. There will be no requirement of a library to execute this code.

Components Required

Arduino Uno + USB Cable

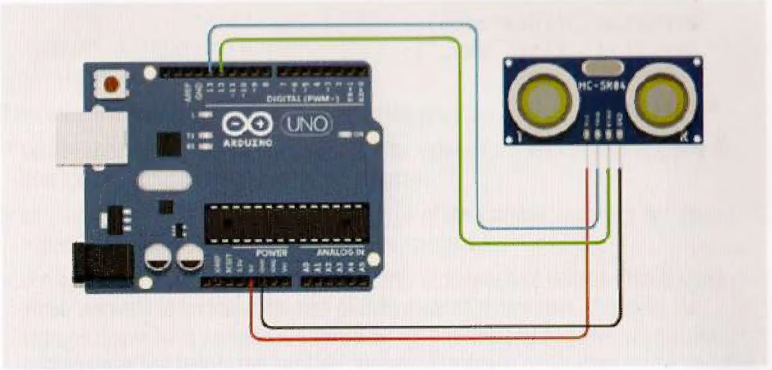
HC-SR04 Ultrasonic Sensor

Jumper wires

Procedure

1. Connect the Arduino and the Ultrasonic sensor using jumper wires as depicted in the diagram.
2. From Tools > Board: Choose "Arduino/Genuino Uno" and then from Tools > Port Choose the port that shows (Arduino/Genuino Uno).
3. Upload the code given overleaf, to the Arduino.

Circuit Diagram

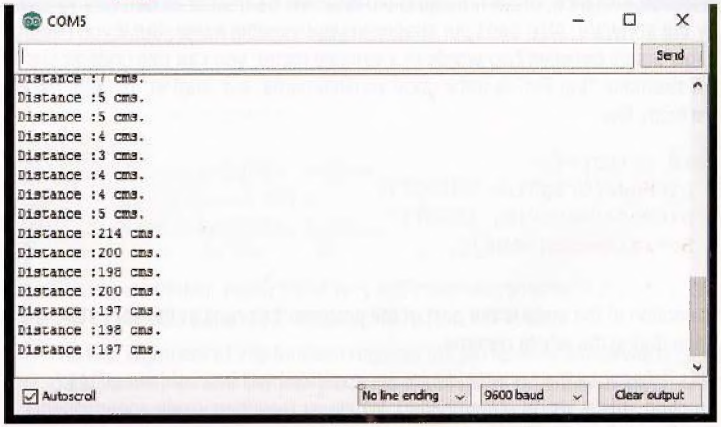


Arduino Pin:	5V	13	12	GND
Ultrasonic Sensor Pin:	VCC	Trig	Echo	GND

Arduino Code

```
const int trigPin = 13;
const int echoPin = 12;

unsigned long duration;
unsigned int distance;

void setup(){
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
}

void loop(){
  digitalWrite(trigPin, LOW);
  delay(1);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.017;

  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cms.");

delay(100);
}
```


4. After the code is done uploading, open up the Serial Monitor by clicking on Tools > Serial Monitor, or by pressing Ctrl+Shift+M on the keyboard or by clicking this botton 



Code Explanation

The code for this program is really simple and short. It basically shows how the Arduino collects data from the sensor. Let's begin:

```
const int trigPin = 13;
const int echoPin = 12;

unsigned long duration;
unsigned int distance;
```

These are the variables that will be used in the program.

- const (short for constant) suggests that the value of the variable is not going to change throughout the course of the program.
- int (short for integer) suggests the data type of the variable, meaning the type of values stored in the variable is going to be integers.
- unsigned suggests that the variable will only store positive values. This is used when we want to increase the size of data a variable can store. Typically the storage space for a variable is split 50:50 for positive and negative values. By putting unsigned before the variable removes the space consumed by the negative values and gives it to the positive values, so that larger numbers can be stored in that variable.
- long is a data type that is used to indicate that the variable is going to store very large numbers as compared to a typical int variable. These numbers are also whole numbers, like the int data type.

- trigPin, echoPin, duration and distance are all variable names. It is always a good idea to name your variables in such a way that it makes sense to the reader about what is stored inside them. Variables in Arduino are case sensitive, meaning that distance, Distance, dIsTaNcE and DISTANCE will be treated as different variables by the software. Also don't put spaces in your variable name. But if you need to show space between two words in a variable name, you can use underscore "_". For example: "trig Pin" is not a good variable name, but "trigPin" or "trig_Pin" is perfectly fine.

```
void setup(){
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
}
```

This section of the code is the part of the program that runs at the very start and only once during the whole runtime.

- This is because the commands are encapsulated within a void setup(){} function.This is worth remembering. Whatever is written inside a void setup() function definition will be executed only once at the very beginning of the program.
- pinMode(pin number, mode) is used to initialise the pins of the Arduino. This means that we're telling the Arduino in advance about the pins that we want to use for our project and which mode we want to use them in.
- The first parameter, or argument in the pinMode() function is the pin number. This can be Digital I/O Pins from 0-13, or Analog Input Pins from A0-A5.
- The second parameter is pin type. And that can be either INPUT, OUTPUT, INPUT_PULLUP.
- INPUT: The pin will accept input data or sensor data.
- OUTPUT: The pin will give output data.
- INPUT_PULLUP: The pin will collect input data, but it will also be connected internally to a resistor that keeps the default state of the pin as HIGH.

Read arduino reference articles for pinMode, INPUT, OUTPUT and INPUT_PULLUP for more detailed information on these topics. To access it, go to Help Reference> which will open a new page in your browser. From there you may look at whichever topics interest you.

We have assigned the trigger pin or pin 13 as output, and the echo pin or pin 12 as input.

- Serial.begin(baudrate) is a command that is used to initialise Serial communication between the Arduino and the computer. This allows both devices to exchange data. 9600 specifies the speed of the data transfer in units of baudrate. The baud rate for a computer can be either 300, 600, 1200, 2400, 4800, 9600,14400, 19200, 28800, 38400, 57600, or 115200.

Here is where the program actually starts interacting with the sensor.

- void loop() is the function that is executed next after void setup() is done. void loop(), as the name implies, is a function that runs in an infinite loop. This function will be continuously executed for the runtime of the program. The loop stops only when the Arduino is powered down or the circuit is reset.

```
void loop(){
  digitalWrite(trigPin, LOW);
  delay(1);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

- digitalWrite(pin number, value) has a very self explanatory name. It is used to output or "write" digital data (0s and 1s) to a specified pin of the Arduino.
  - The first argument of this function requires the pin number which will output the data. Do keep in mind that this pin number should be initialized using pinMode() in the void setup() function.
  - The second argument of this function is where you can specify the output of the pin, whether it will be HIGH or LOW
    - HIGH: This state means that the pin will output 5V DC and upto 25mA of current. This is also known as sourcing current.
    - LOW: This state means that the pin will act as a GND pin and will absorb upto 25mA of current. This is also called sinking current.

In the program, the trigger pin is kept LOW initially. Meaning that there is no pulse being generated by the sensor. After a delay of 1 millisecond, the trigger pin is taken to a HIGH state by using a digitalWrite statement, kept high for 10 microseconds by using the delayMicroseconds() function and finally, again taken to a LOW state by using another digitalWrite() statement.

Here is what happens due to that small section of code. A 10 microsecond long pulse is generated. Which is made up of a 40KHz Ultrasonic Square wave.This pulse exits the sensor and hits any obstacle in its path. In the next section of the code, we will detect the reflected pulses and calculate the distance.

```
duration = pulseIn(echoPin, HIGH);
distance = duration * 0.017;
```

In this section of the code, as mentioned before, we record the reflected pulses and calculate distance.

- pulseIn(pin number, value) is a function that is basically used as a stopwatch. It can measure the time it takes for a pin to change its state from HIGH to LOW or LOW to HIGH. It has two arguments, pin number and value.
  - Pin number specifies the pin from which the pulse is being read. Could be between 0-13 or A0-A5.
  - Value specifies if the pulse is starting from LOW or HIGH. Suppose if the pulse is starting HIGH, so the function will wait for the pin to go HIGH from LOW, start recording the time in milliseconds, and then stop recording when the pin goes LOW again.
- Okay so in this section of the code, we are reading the pulses that are being recorded by the echo Pin. The duration of the pulses is recorded in milliseconds.

Now, we know the speed of sound in air is 340m/sec and we are recording the time taken for sound to travel to the obstacle and back again. Since we have the time and the speed, we can calculate distance. This is how we achieve that:

Distance travelled by the sound wave(D) = Speed of sound in air(V) x Time measured by the Arduino T/2

If you are wondering why time is being divided by two, consider the fact that the sound is travelling the distance between the obstacle and the sensor twice: once while going towards the obstacle and the second time while being reflected back towards the sensor. This means that the arduino is recording the time taken by the whole roundtrip, while we need the time required for one way travel only. Hence it is divided in half.

Now, we want the distance in centimetres, and the time is being measured in microseconds. So making the conversions to make the unit cm/second, we get:

$D=(V \times 100)/1000000 \times T/2$

Substituting value of V with the speed of sound in air: 340 m/sec:

$D=(340 \times 100)/1000000 \times T/2$

Hence we get: $D=T \times 0.017$

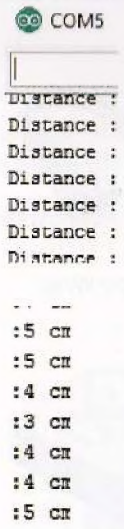
Which is the formula mentioned by this line: distance = duration * 0.017;

Lastly, we will look at how the Arduino displays the values over the Serial Monitor.


```
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cms.");

delay(100);
}
```

This is the last section of the code. In this section you will see how you can use Serial to send data to the computer. This is a feature that is very useful especially while prototyping since it helps in debugging and reporting the state of a program while running. It is also useful for data collection and analysis. Let's take a look at the different commands used here.

- Serial.print() is used when a user wants to display something on the Serial monitor on the computer. Notice that there is two different types of arguments within the Serial.print() function. One is "Distance: " and the other is distance. This is the difference between the two:



  - When we want to display any number, text, special character, or message as it is on the screen, we can put it within the double inverted commas. Hence the statement Serial.print("Distance: "); produces this output:
  - The second type of argument is when we want to show changing values or variables on the screen. In this case, we can directly put the name of the variable in the argument of the Serial.print() function. Hence the statement Serial.print(distance); produces this output:

That's it. You can make this program more complex by adding commands to display the distance in different units. And you can also change the program to calculate velocity. Try to modify the code in such a way that you can calculate the velocity of objects in front of the sensor. Also assign a sign to the velocity based on direction.

3. Blink Radar

Project 2

This example will speed up or slow down the blinking of an LED based on the distance of an obstacle from the sensor. The New Ping library, made byTim Eckel, will be used in this code.

Components Required

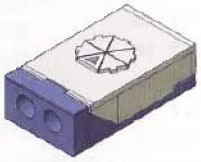
Arduino UNO + USB Cable

HC-SR04 Ultrasonic Sensor

LED (Any color will do)

Jumper Wires

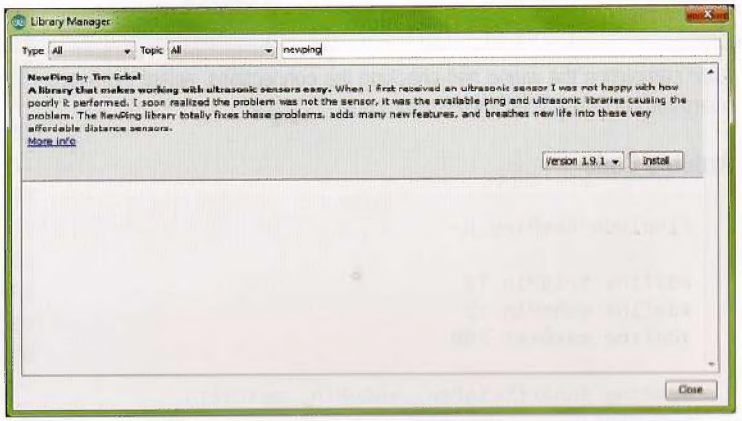
9V Battery + Battery Clip



3D Printed parts
(Optional)

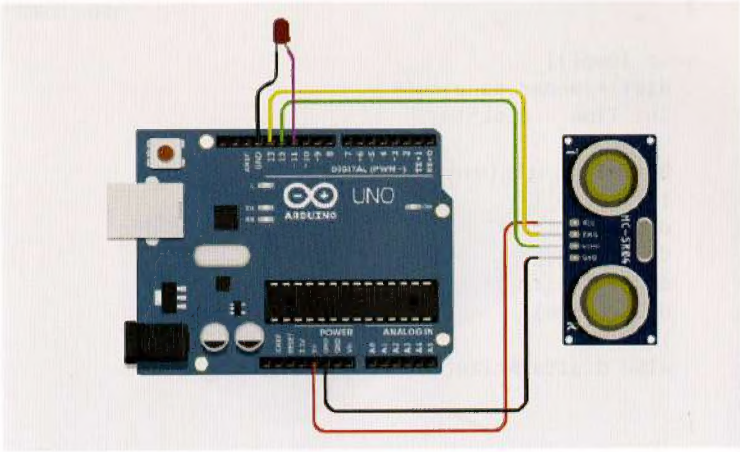
Installing Library

To install the NewPing Library for Arduino, go to Tools>Manange Libraries, which will open up the Library manager. In the search bar, type New Ping and wait for the result. This is what you should see: Here you can see that the library is called NewPing and is made by Tim Eckel. Select the latest version of the library and click on install in the lower right corner of the box. Note that you will need a working internet connection for this to work.



After the library is done installing, close the Library Manager. We'll now connect the circuit according to the circuit diagram given below.

Circuit Diagram



Arduino Pin:	5V	13	12	GND
Ultrasonic Sensor Pin:	VCC	Trig	Echo	GND

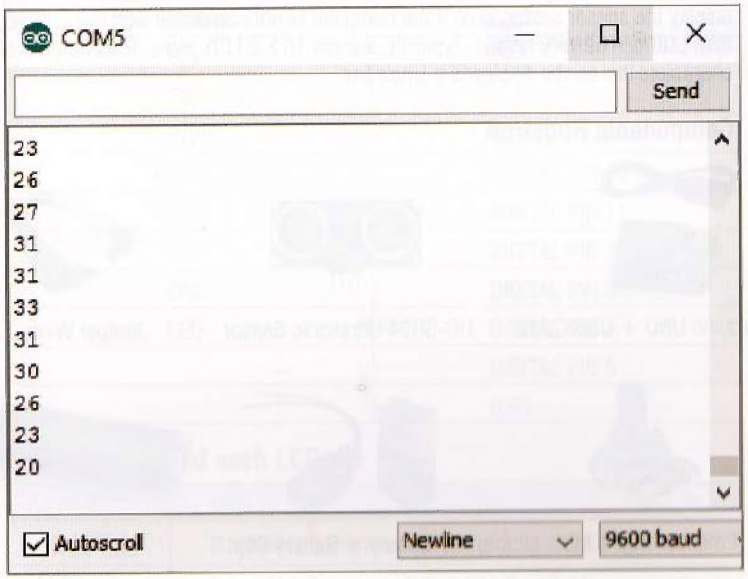
Arduino Pin:	11	GND
LED Pin:	Anode (Positive/Longer Pin)	Cathode (Negative/Shorter Pin)

After completing the wiring and checking the connections, select the appropriate board and port and upload the following code:

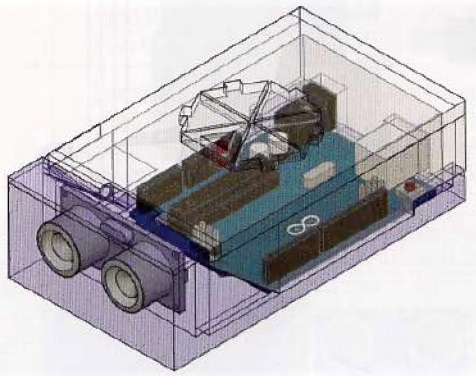
Arduino Code

```
#include<NewPing.h>

#define trigPin 13
#define echoPin 12
#define maxDist 200

NewPing sonar(trigPin, echoPin, maxDist);
int dist;

void setup(){
  pinMode(11, OUTPUT);
  Serial.begin(9600);
}

void loop(){
  dist = sonar.ping_cm();
  int Time = dist*10;

  Serial.println(dist);
  if(Time>0){
  digitalWrite(11, HIGH);
  delay(Time);
  digitalWrite(11, LOW);
  delay(Time);
  }
  else digitalWrite(11, LOW);

}
```


Once you're done uploading, you should be able to see the LED Blink based on the distance of an obstacle from the sensor. You can also see the following sensor output on the Serial Monitor:



This is an optional step, but you can enclose your project in a custom-made 3D Printed case.



4. Ultrasonic Sensor with LCD

Project 3

When we are working with a physical arduino and ultrasonic sensor and if we want to display the sensor setting even if the computer is not connected with the arduino, then LCD can be very helpful. Typically, we use 16 X 2 LCD. Here, 16 is number of characters that can be displayed in single line

Components Required

Arduino UNO + USB Cable

HC-SR04 Ultrasonic Sensor

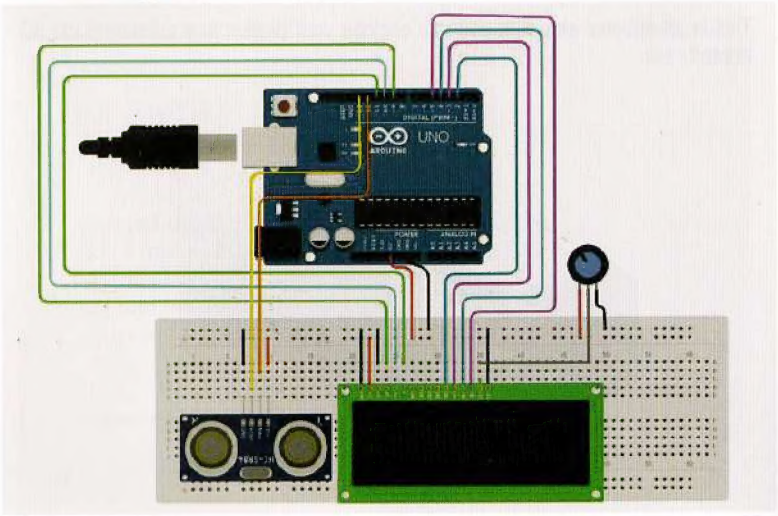
Jumper Wires

Potentiometer 1 kΩ

9V Battery + Battery Clip

LCD 16 X 2

Circuit Diagram



Connection of LCD with Arduino: Connect each pin on LCD with relevant pins of Arduino.

LCD	Arduino
VssVcc	GND
Vee	5V
RS	GND
R/WE	DIGITAL PIN 9
Db4	DIGITAL PIN 10
Db5	DIGITAL PIN 11
Db6	DIGITAL PIN 2
Db7	DIGITAL PIN 3
LED-	DIGITAL PIN 4
	DIGITAL PIN 5
	GND

Let us understand each LCD pin

Name of pin	Use
Vss, LED-5	It provides common negative/ground to the LCD
V, LED+	It provides power to LCD and its Backlight
Vee	Controls the contrast of LCD
RS	Register Select: Selects command register when low, and data register when high
R/W	Read/Write - Low to write to the register; High to read from the register
E	Enable - Sends data to data pins when a high to low pulse is given; Extra voltage push is required to execute the instruction and EN(enable) signal is used for this purpose. Usually, we set en=0, when we want to execute the instruction we make it high en=1 for some milliseconds. After this we again make en=0.
DB0 To Db7	8-bit Data Pins
LED+	Backlight LED Positive
LED-	Backlight LED Negative

Connection of Potentiometer: Potentiometer is used to set the correct resistor value in the circuit. After connecting all circuit components and uploading the program, if a message is not getting displayed on the screen then adjust the knob of the potentiometer.

Potentiometer	
Terminal 1	Arduino 5V
Terminal 2	LCD LED+
Terminal 3	Arduino GND

Connection of Ultrasonic Sensor with Arduino

Ultrasonic Sensor	Arduino
Vcc	5V
GND	GND
Trigger	DIGITAL PIN 12
Echo	DIGITAL PIN 13

Arduino Code

```
#include <LiquidCrystal.h>
LiquidCrystal LCD(9,10,11,2,3,4,5);

#define trigPin 12
#define echoPin 13
```

Here, first we have installed the libraries of LCD. You can find it in "sketch" menu "Include Library" option

We need to define that on which digital pins we have connected our ultrasonic sensor. So, we have defined two variables called trigPin and echoPin and we have assigned them pin numbers. Remember that arduino programming language is case sensitive so if you are defining any variable, you need to use the same name everywhere in the program without any change in capital or small letters.


```
void setup()
{
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  LCD.begin(16,2);
  LCD.setCursor(0,0);
  LCD.print("Target Distance:");
}
```

Those codes which we want to execute once are written inside void setup(). First, we need to define the modes of pins on which an ultrasonic sensor is attached. Trigger pin is used to emit ultrasonic sound and echo pin catches if the sound wave is reflected. So, we will define Trigger pin as OUTPUT and Echo pin as INPUT.

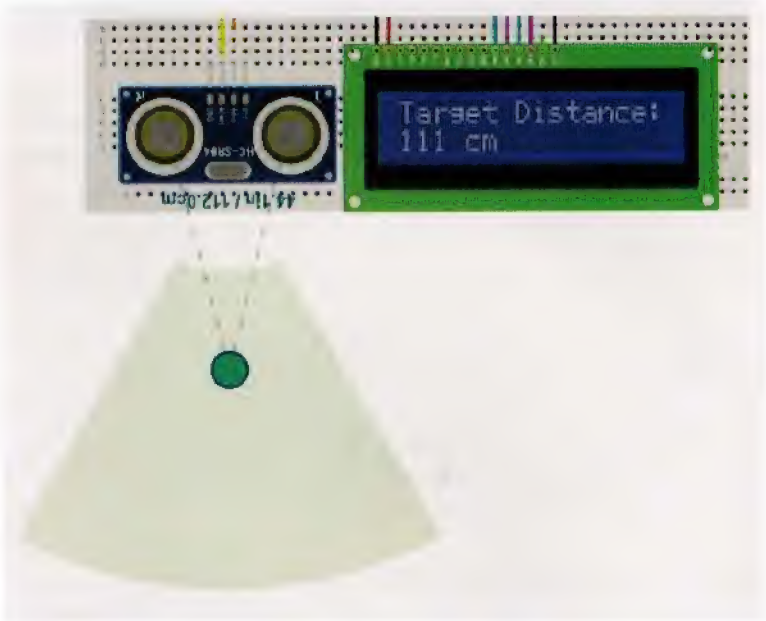
The LCD that we are using is 16 X 2. That means it can display messages in 2 rows where each row can have a maximum 16 characters. The Cursor of the LCD is set to (0,0) which will be in the top left corner and in the first row, the message is written as "Target Distance".

```
void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;

  LCD.setCursor(0,1);
  LCD.print("                ");
  LCD.setCursor(0,1);
  LCD.print(distance);
  LCD.print(" cm");
  delay(250); //pause to let things settle
}
```

In void loop(), the rest of the code is written. We will start with triggering the Trigger pin of the ultrasonic sensor and then catching the echo of the sound wave on Echo pin using pulseIn. Then, we will calculate the distance using the formula of speed (for explanation of this formula, refer project-1). Depending on which unit you want to display your distance in, the multiplication factor of distance formula will change.

Then, we set the cursor in the 2nd row and printed the distance. Here, note that when we write LCD.print("some message") then the message will get printed as it is because it is written in the "". If you want to print the value of any variable (in our case distance), you need to write it without "".

Once, we upload this program in Arduino, the distance starts displaying on the LCD screen. If you don't get any message then try adjusting the potentiometer a little bit. After testing the system with computer, you can power up the Arduino using a 9V battery so that it can work without laptop of computer as individual system.



Application

Here are some examples of projects that you can do using this circuit.

1. Automatic Height Measuring Device
2. Automatic Parking System in Vehicle
3. Vehicle Collision Detection System

5. Other Project Ideas - Ultrasonic Sensor

Following are some project ideas that can be prepared using Ultrasonic Sensor.

A. Smart Walking Stick for visually challenged

- With the distance measuring capacity of ultrasonic sensors, it can be very helpful in making a smart walking stick for visually challenged people.
- Visually challenged people often face problems in detecting obstacles in front of them. You can build a small device that can be fitted in their stick.
- The device should consist of an ultrasonic sensor, arduino, buzzer or vibrating motor.
- On detecting the obstacle, it will either make a buzzer sound or it can produce some vibration in the stick.
- To make this device in a smaller space, you can use an arduino nano.
- Refer to this link for detailed description of this project, circuit diagram and arduino code.

https://www.circuitstoday.com/ultrasonic-blind-walking-stick

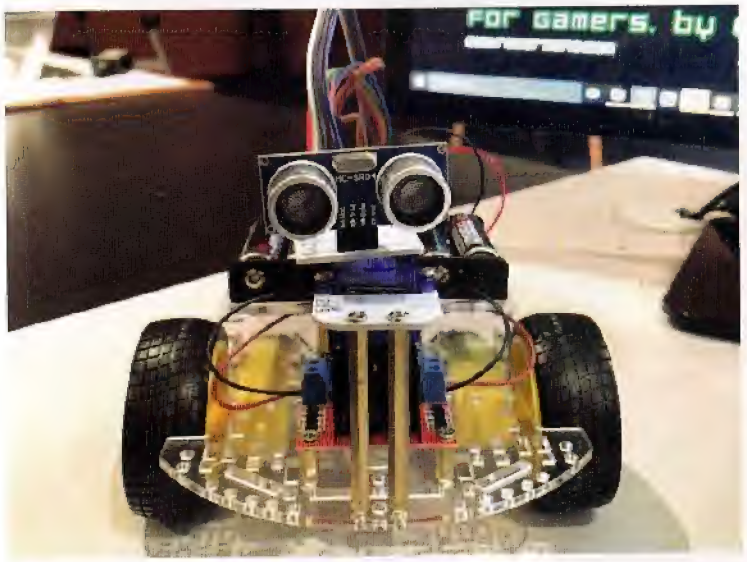
B. Smart Dustbin

- Smart dustbin is one of the very popular projects. Everytime when any person wants to throw garbage in the dustbin, he/she has to touch the top part of the dustbin, open it and throw garbage inside.
- In public areas, sometimes people don't bother to do that. Sometimes touching the dustbin is not good for hygiene purposes.
- So, you can create a dustbin that has an ultrasonic sensor. When a person comes nearer to the dustbin, it senses the distance and it opens the dustbins using a servo motor.
- Additionally, you can also create a system to sense whether the dustbin is full or not. This will help in monitoring garbage collection systems especially in public places.

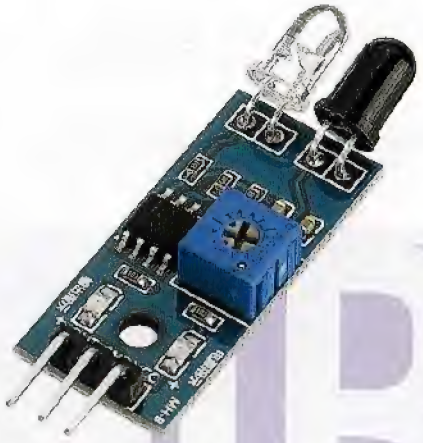
https://www.instructables.com/Smart-Dustbin/

C. Robot Navigation System

- When any robot is moving in a confined space like inside a room, an ultrasonic sensor can be very helpful in determining path for moving.
- Here, the concept is the same as obstacle avoiding robots but using multiple ultrasonic sensors to sense obstacles in multiple directions can improve the path planning of your robot.
- In this project, you need to create an algorithm which determines the distance of obstacle from a robot and then gives command to motors to rotate accordingly
- First start with single sensor and then increase the complexity of the project.



https://create.arduino.cc/projecthub/ishaq-yang/auto-ultrasonic-car-a85c6f



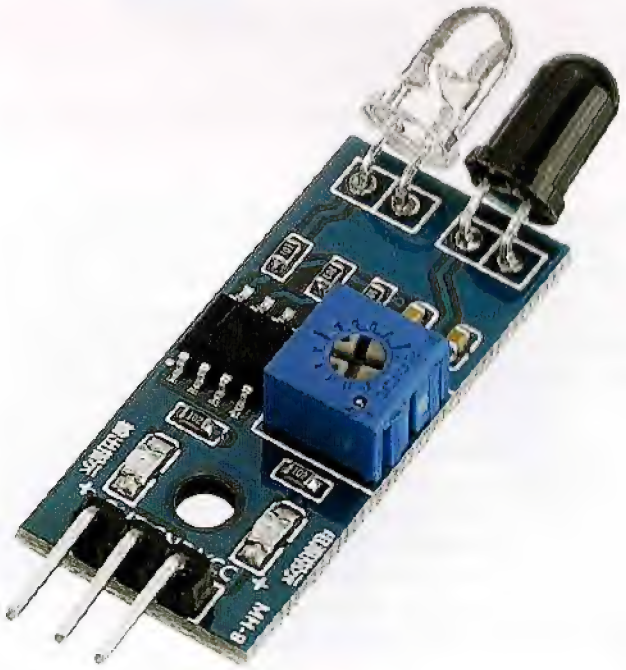
IR Sensor

6. Getting to know IR Sensor

Electronic gadgets are made from various sensors which are derived and inspired from nature. We will make a few projects using Infrared Sensor (IR Sensor) and also understand how it is used to sense the surrounding environment.

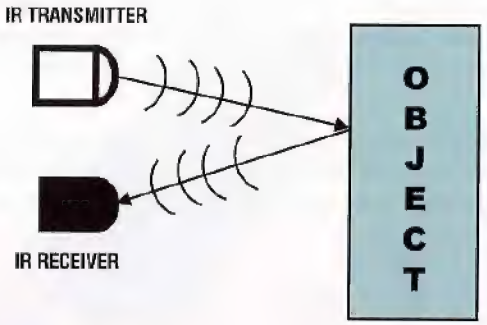
An IR Sesnor 'sense' the surrounding environment such as by detecting the heat of an object or its motion. Some IR sensors detect objects' temperatures by measuring the amount of infrared radiation that is naturally emitted by the objects.

Other devices detect the presence of objects by emitting their own IR radiation forward and then measuring the amount reflected back. It is used in security cameras, remote control devices, Thermal imaging technology and many more.



7. Concept of IR Sensor

The basic concept of an Infrared Sensor which is used as an obstacle detector is to transmit an infrared signal, this infrared signal bounces from the surface of an object and the signal is received at the infrared receiver.



Types of IR Sensors

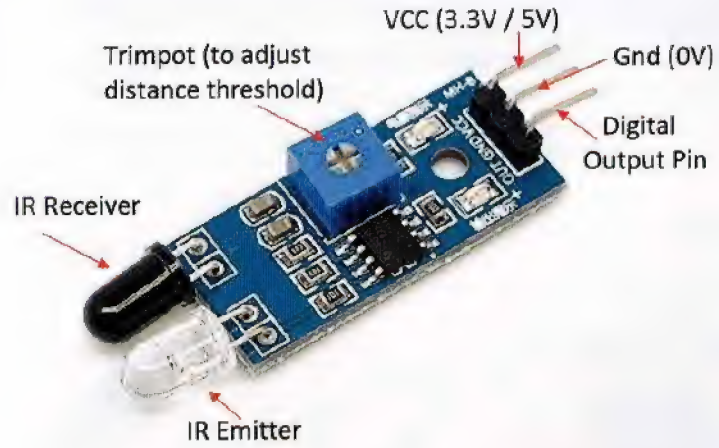
Infrared sensors can be passive or active. Let's understand more about active IR Sensors

Active infrared sensors consist of two elements: infrared source and infrared detector. Infrared sources include an LED or infrared laser diode. Infrared detectors include photodiodes or phototransistors. The energy emitted by the infrared source is reflected by an object and falls on the infrared detector.

It is a digital type sensor so you can get the sensor value as either 0 or 1. Typically, the range of IR sensor modules shown in the above image is 7 cm.

IR Sensor Pin Information

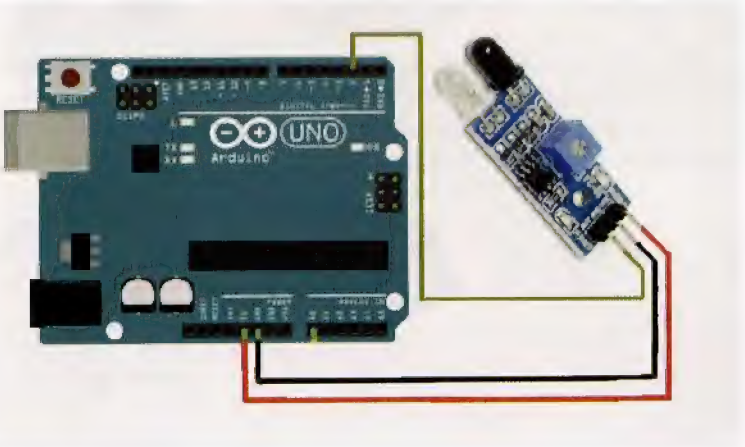
IR sensor module consists of an IR led (IR Emitter), a photodiode (IR Receiver), a potentiometer (Trimpot), an IC Operational amplifier, and an LED. IR led emits the infrared light whereas photodiode detects the infrared light. A potentiometer is used to change the range of detection and calibrate the output of the sensor according to the requirement. IR sensors can be used with any Arduino board like nano, UNO, mega. Input voltage to the sensor should be 3V to 5V DC.



Connection of single IR sensor module with Arduino

Pin Name	Description
VCC	Arduino 5V
GND	Arduino GND
OUT	Digital pin 2 to 13

Circuit Diagram



Arduino Code

Open the Arduino IDE software on your computer. Open the new sketch File by clicking New. Enter the following code. This is the basic program to get the sensor value in Serial Monitor

```
int irsensor = 0;

void setup() {

 pinMode(5,INPUT);                    // sensor pin
connected with Arduino pin 5
 Serial.begin(9600);
}

void loop() {

 irsensor = digitalRead(5);          // reading sensor
data
 Serial.println(irsensor);             //printing sensor
data in serial monitor
}
```

This program will simply read the value of the IR sensor and show it in the Serial Monitor. (Serial monitor can be opened from the Tools menu or the shortcut is Ctrl + Shift + M.

If there is any obstacle in the proximity of the sensor then IR light will get reflected and sensor value will show 1 as OUTPUT or else it will show 0 as OUTPUT.

Let us make some simple projects using IR sensor, this will also help us in understanding the working of IR sensor.

8. Line Follower Robot

Project 4

IR sensors can be used to make a line follower robot. A simple physics phenomenon of white color reflecting light completely and black color absorbing light completely is used here.

Components Required

IR Sensor Module

Jumper wires

Arduino UNO + Cable

9V Battery and Snapper – 1

Wheels

DC Motors

Chassis - Robot body

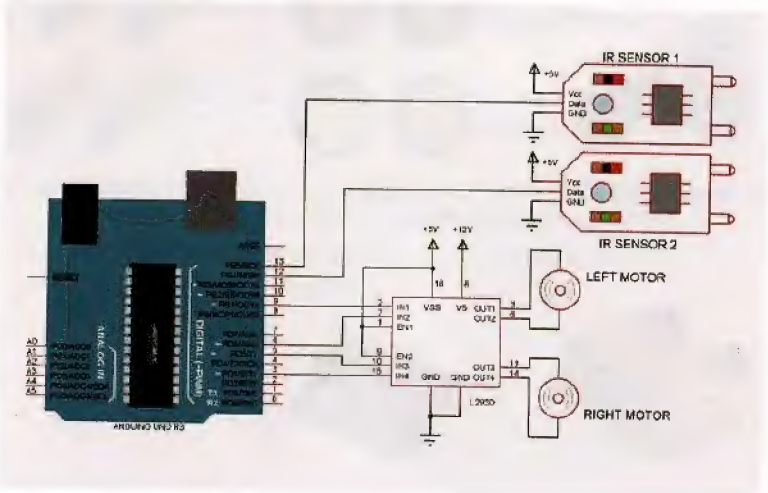
L293D Motor Driver IC

Sr No.	Component Name and specification	Qty.
1	IR Sensor Module	2
2	Arduino UNO + Cable	1
3	L293D Motor Driver IC	1
4	DC Geared Motors, 300RPM	4
5	Wheels	4
6	Chassis - Robot body	1
7	9V Battery + Snapper	1
8	Jumper wires	As per requirement

Circuit Diagram

Take care of following points during making of the circuit

- 5V and GND pins of the IR sensor can be connected with 5V and GND pins of Arduino. Use a breadboard to make common 5V and GND connections.
- Do not connect the motor directly with Arduino. Use L293D motor driver IC to connect motors.



[Reference: https://create.arduino.cc/projecthub/Amrendra0110/line-follower-robot-026dcd]

Arduino Code

```
#define LS 2 // left sensor
#define RS 3 // right sensor
#define LM1 5 // left motor M1a
#define LM2 4 // left motor M2a
#define RM1 7 // right motor M2a
#define RM2 6 // right motor M2b

void setup()
{
  pinMode(LS, INPUT);
  pinMode(RS, INPUT);
  pinMode(LM1, OUTPUT);
  pinMode(LM2, OUTPUT);
  pinMode(RM1, OUTPUT);
  pinMode(RM2, OUTPUT);

}

void loop()
{
if(digitalRead(LS) && digitalRead(RS)) // Move Forward
on line

   {

      digitalWrite(LM1, HIGH);
      digitalWrite(LM2, LOW);
      digitalWrite(RM1, HIGH);
      digitalWrite(RM2, LOW);

   }

if(digitalRead(LS) && !(digitalRead(RS))) // turn left
by rotating left motors in forward and right ones in
backward direction

   {
```

```
 digitalWrite(LM1, HIGH);
      digitalWrite(LM2, LOW);
      digitalWrite(RM1, LOW);
      digitalWrite(RM2, HIGH);
   }

if(!(digitalRead(LS)) && digitalRead(RS)) // Turn right
by rotating right motors in forward and left ones in
backward direction

   {

      digitalWrite(LM1, LOW);
      digitalWrite(LM2, HIGH);
      digitalWrite(RM1, HIGH);
      digitalWrite(RM2, LOW);

   }


if(!(digitalRead(LS)) && !(digitalRead(RS))) // Finish
line, stop both the motors

   {

      digitalWrite(LM1, LOW);
      digitalWrite(LM2, LOW);
      digitalWrite(RM1, LOW);
      digitalWrite(RM2, LOW);

   }

}
```

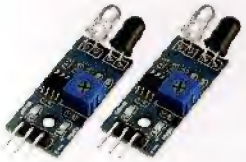

9. Visitor Counter - Bidirectional

Project 5

In this project we will use the IR sensors to detect how many people have entered a room and how many people have left the room. This system can be used in theatres, classrooms during any ceremony or functions.

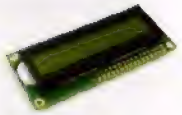
It works on the simple principle of reflection of light. When a person passes through the gate, infrared light will get reflected and received by the sensor thus, counting the person. 2 sensors are kept in the system so if sensor-1 detects the person first and sensor-2 detects second then the person is going inside and if the sensing sequence is reversed then the person is coming outside.

Components Required



2 IR sensors

Arduino UNO + Cable

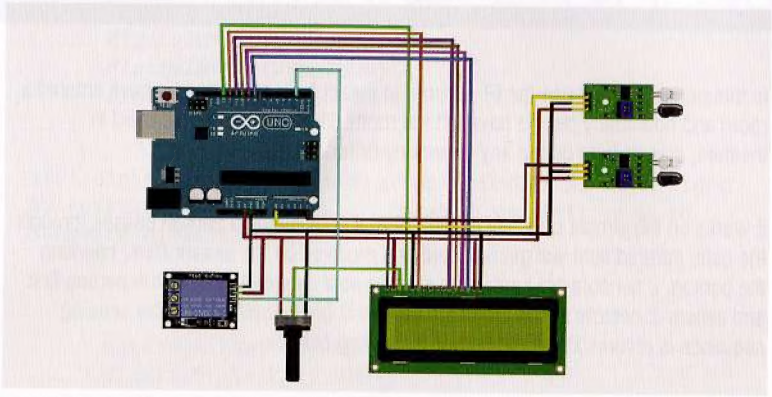


LCD display

Jumper wires

Relay module

Circuit Diagram



[https://www.creativitybuzz.org/visitor-counter-using-arduino/]

Arduino Code

```
#include<LiquidCrystal.h>

LiquidCrystal lcd(13,12,11,10,9,8);

define in 14
define out 19
define relay 2
int count=0;

void IN()
{
   count++;
   lcd.clear();
   lcd.print("Person In Room:");
   lcd.setCursor(0,1);
   lcd.print(count);
   delay(1000);
}
void OUT()
{
   count-;
   lcd.clear();
   lcd.print("Person In Room:");
   lcd.setCursor(0,1);
   lcd.print(count);
   delay(1000);
```



```
}
void setup()
{
   lcd.begin(16,2);
   lcd.print("Visitor Counter");
   delay(2000);
   pinMode(in, INPUT);
   pinMode(out, INPUT);
   pinMode(relay, OUTPUT);
   lcd.clear();
   lcd.print("Person In Room:");
   lcd.setCursor(0,1);
   lcd.print(count);
}
void loop()
{
   if(digitalRead(in))
      IN();
   if(digitalRead(out))
      OUT();
   if(count<=0)
   {
      lcd.clear();
      digitalWrite(relay, LOW);
      lcd.clear();
      lcd.print("Nobody In Room");
      lcd.setCursor(0,1);
      lcd.print("Light Is Off");
      delay(200);
   }
   else
   {
      digitalWrite(relay, HIGH);
   }

      lcd.print("Nobody In Room");
      lcd.setCursor(0,1);
      lcd.print("Light Is Off");
      delay(200);
   }
   else
   {
      digitalWrite(relay, HIGH);
   }
```

10. Speed Tracker

Project 6

Using an IR sensor, you can make a speed tracker for cars. 2 IR sensors are placed at a distance. When a car or any vehicle passes, the 1st IR sensor detects the car and then the 2nd IR sensor detects it, the controller measures the time between both detection and based on speed-distance-time formula, the speed of the car is measured.

Components Requirement

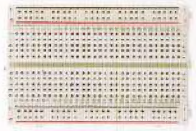
Sr No.	Component Name	Qty
1	IR Sensor Module	2
2	Arduino UNO + Cable	1
3	LCD 20 X 2 with I2C Module	1
4	LEDs	1
5	Breadboard	1
6	9V Battery + Snapper	1
7	Jumper wires	As per requirement

2 IR sensors

Arduino UNO + Cable

LED

LCD 20 X 2 with I2C Module

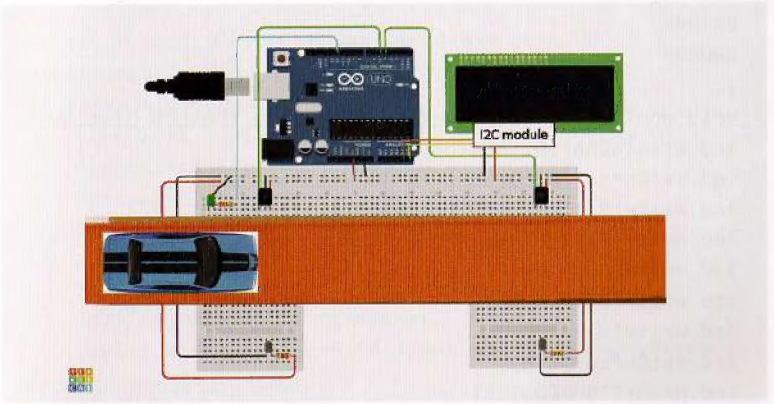


Breadboard

9V Battery and Snapper

Jumper wires

Circuit diagram



https://create.arduino.cc/projecthub/NerdFatherRJ/speeduino-speed-tracker-c8fcae

Arduino Code

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27,20,4);// I2C adress
const int ledPin = 12;

byte irPinA = 4;
byte irPinB = 5;
byte irValA;
byte irValB;
float diff;
float vel; // calculated speed
unsigned long timeFirst;// IR sensor at irPinA
unsigned long timeScnd;// IR sensor at irPinB
float speedConst = 1080; // ((distance between IR
sensors in mm)(300mm x
3600)/1000)to convert mm/millis to km/h)

byte customChar0[8] = { //some art for the LCD screen
B01000,
B01100,
B01110,B01111,
B01110,
```

```
B01100,
B01000,
B00000
};
void configLCD(){ //function to config the LCD display
lcd.createChar(0, customChar0);
lcd.setCursor(0,0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.setCursor(5,0);
lcd.print("SPEED TEST");
lcd.setCursor(15,0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.write(0);
lcd.setCursor(0,1);
lcd.print("P1:");
lcd.setCursor(0,2);
lcd.print("P2:");
lcd.setCursor(0,3);
lcd.print("Speed:");
}
void setup()
{
pinMode(irPinA, INPUT);
pinMode(irPinB, INPUT);
pinMode(ledPin, OUTPUT);
lcd.init();
lcd.backlight();
configLCD();
digitalWrite(ledPin, HIGH); //Green LED HIGH: ok to
next ride.
}
void loop()
```



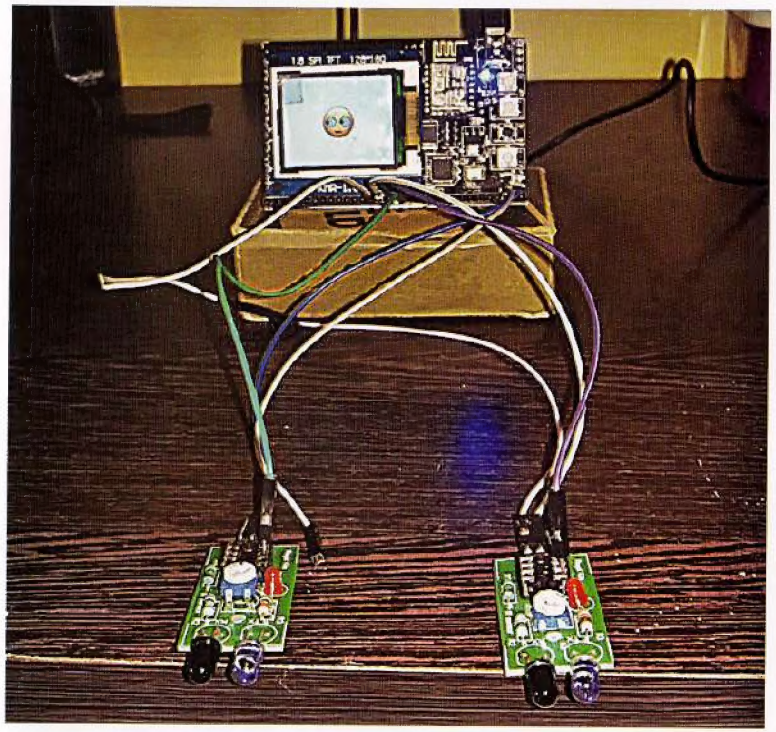
```
{
irValA = digitalRead(irPinA);irValB =
digitalRead(irPinB);
if (irValA == LOW){
    timeFirst = millis();
    digitalWrite(ledPin, LOW);
    delay(30);
    }
if (irValB == LOW){
    timeScnd = millis();
     diff = timeScnd - timeFirst;
     vel = speedConst / diff;//get the Speed converted
from mm/millis to km/h.
    /*
    use only if you want
    Serial.print("T1: ");
    Serial.println(timeFirst);
    Serial.print("T2: ");
    Serial.println(timeScnd);
    Serial.print("Diff: ");
    Serial.println(diff);
    Serial.print("Speed: ");
    Serial.println(vel);
    */
    lcd.setCursor(6, 1);
    lcd.print(timeFirst);
    lcd.setCursor(6, 2);
    lcd.print(timeScnd);
    lcd.setCursor(6,3);
    lcd.print(vel);
    lcd.print("km/h");
    delay(5000);
     lcd.clear();
    configLCD();
     digitalWrite(ledPin, HIGH);
    }
}
```

11. Other Project Ideas - IR Sensor

Following are some project ideas that can be prepared using IR Sensor.

A. Detecting Swipe Motion

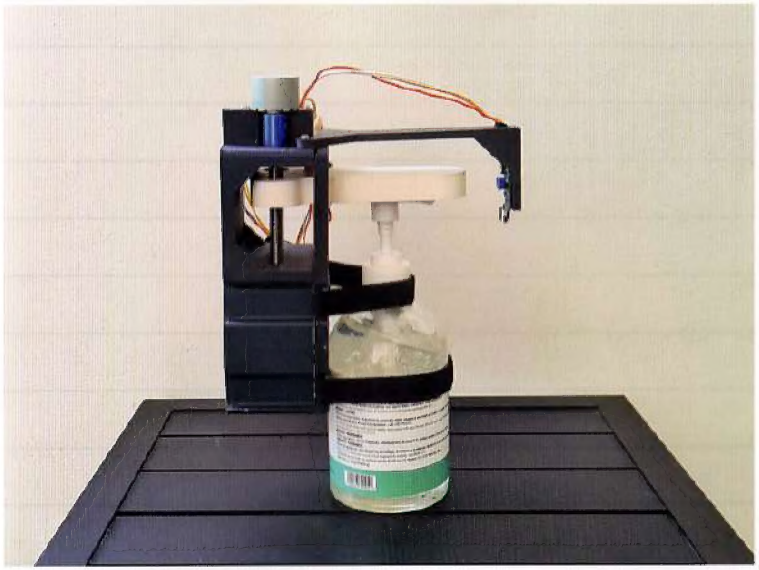
IR sensors can be useful in detecting right-left swipe. In this project, you will need two IR sensors. When we swipe left, we move our hand from right to left. The right-IR sensor detects this movement first. Now only, if any movement is detected at the left-IR sensor it recognizes it as a left swipe. Similar for right swipe as well. It will only detect the swipe when both IR sensors detect the hand one by one. Check out the reference for further information.



[https://www.instructables.com/Simple-Gesture-Control-Using-IR-Sensors/]

B. Touch Free Sanitiser Dispenser

IR sensors can be used to make automatic hand wash or automatic water tap projects. When the sensor detects the hand kept under tap or hand wash, it will give a signal to the actuator used in the system to open the tap or pull the string to let washing soap come out of the bottle. These projects can be done in a variety of ways using different sensors and actuators (motors). Here are some examples



1.https://medium.com/@anikamimi47/how-to-make-automatic-touchless-hand-wash-addb813c118b
2.https://create.arduino.cc/projecthub/Nikolas550/automatic-hand-sanitizer-c22fcc
3.https://create.arduino.cc/projecthub/MissionCritical/diy-hand-sanitizer-dispenser-using-arduino-143de1